Topology-based Representations for Motion Planning and Generalisation in Dynamic Environments with Interactions

Vladimir Ivan*, Dmitry Zarubin†, Marc Toussaint†, Taku Komura*, and Sethu Vijayakumar*

*School of Informatics, University of Edinburgh, UK
†Department of Computer Science, FU Berlin, Germany

June 15, 2012

Abstract

Motion can be described in several alternative representations, including joint configuration or end-effector spaces, but also more complex *topology-based* representations that imply a change of Voronoi bias, metric or topology of the motion space. Certain types of robot interaction problems, e.g. wrapping around an object, can suitably be described by so-called writhe and interaction mesh representations. However, considering motion synthesis solely in a topology-based space is insufficient since it does not account for additional tasks and constraints in other representations. In this paper, we propose methods to combine and exploit different representations for synthesis and generalization of motion in dynamic environments. Our motion synthesis approach is formulated in the framework of optimal control as an approximate inference problem. This allows for consistent combination of multiple representations (e.g., across task, end-effector and joint space). Motion generalization to novel situations and kinematics is similarly performed by projecting motion from topology-based to joint configuration space. We demonstrate the benefit of our methods on problems where direct path finding in joint configuration space is extremely hard whereas local optimal control exploiting a representation with different topology can efficiently find optimal trajectories. In real world demonstrations, we highlight the benefits of using topology-based representations for online motion generalization in dynamic environments.

1 Introduction

Many relevant robotic tasks concern close interaction with complex objects. While standard motion synthesis methods describe motion in configuration space, tasks that concern the interaction with objects can often more appropriately be described in representations that reflect the interaction more directly. For instance, consider the wrapping of arms around an object, e.g. embracing a human. Described in joint space, such a motion is complex and varies greatly depending on the embraced object. When describing the motion more directly in terms of the interaction of arm segments with object parts (e.g. using the interaction mesh representation that we will introduce below) we gain better generalization to objects of different shape or position and online adaptation to dynamic objects. Similar arguments apply to other scenarios, e.g. multi-link articulated robots reaching through small openings and complex structures,

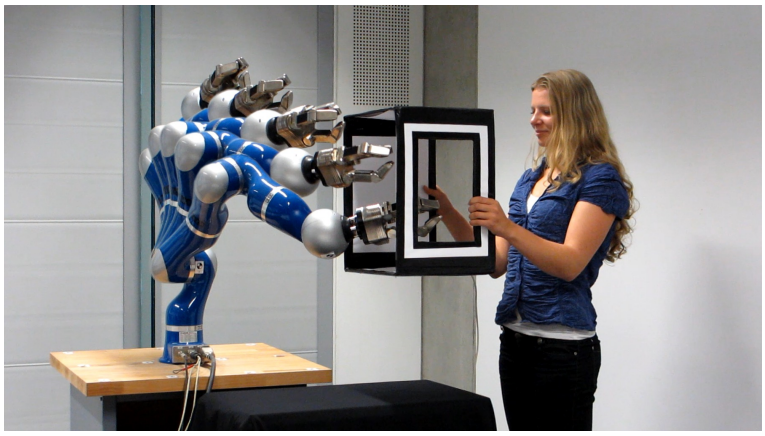


Figure 1: KUKA LWR 4 robotic arm reaching through a hollow box with task being defined in combined Writhe and interaction mesh space, showing an example of planning and dynamic remapping using topology-based representations as described in Section 5.3.

surfaces wrapping around objects and fingers grasping objects. In such cases, the alternate representations greatly simplify the problems of motion generalization as well as planning.

There are several formal views on the implication of an alternate abstract representation: 1) In the context of randomized search, such representations alter the Voronoi *bias* or more generally the sampling strategy and therefore, the efficiency of RRT or randomized road maps. [14] demonstrate this effect in the case of RRTs. 2) An alternate representation may imply a different *metric*, such that a trajectory that is a complex path in one space becomes a simple geodesic in another. 3) An alternate representation may change the topology of the space such that local *optimization* in one space is sufficient for finding a solution whereas global optimization (randomized search) would be needed in the other. 4) Finally, different representations may allow us to express different *motion priors*, for instance, a prior preferring "wrapping-type motions" can be expressed as a simple Brownian motion or Gaussian process prior in one space, whereas the same Brownian motion prior in configuration space renders wrapping motions extremely unlikely.

In this paper, we propose methods that can combine the different representations at the abstract as well as the lower (execution) level and plan simultaneously at these different levels to efficiently generate optimal, collision free motion that satisfy constraints. Each representation has their own strength and weaknesses and coupling them can help solve a wider range of problems. More specifically, our contributions are:

- The introduction of topology-based *representations* tuned to the domain of robot motion synthesis and manipulation, with a strong focus on interaction with and manipulation of the environment.
- A principled extension of a stochastic optimal *control framework* that admits the capability to combine various representations for motion synthesis. This is expressed in a graphical model that couples motion priors at different levels of representations.
- A method for motion *generalization* (remapping) to novel situations via a topology-based representation.
- Experiments that *validate* the benefit of Bayesian inference planning with topology-based representations in problems involving complex interactions.

In the context of this paper, we define the term *topology-based space* as any space that in general abstracts away the geometric detail of work space. Not all the topology-based spaces , therefore, strictly relate to a space with a novel topology — most of these spaces are in fact $\mathbb{R}^n$ with the usual topology and they can be classified as universal covering spaces [16] of the joint space. For example, the writhe scalar described in Section 2.1 is a homology invariant (see Section 2.1.2) but the interaction mesh space defined in Section 2.2 is a metric space designed to represent interaction between objects and to abstract away their geometric detail.

As opposed to the computer animation domain, where topology-based representation have recently been used [8], synthesizing motion in such abstract spaces for planning and control of robotic systems come with additional challenges. Typically, control tasks are specified in world (or end-effector) coordinates, the obstacles may be observed in visual (or camera) coordinates, and the joint limits of the actuators are typically described in joint coordinates. Therefore, the general challenge is to devise motion synthesis methods that combine the benefits of reasoning in topology-based coordinates while preserving consistency across the control coordinates and managing to incorporate dynamic constraints from alternate representations seamlessly.

Here we extend our previous work presented in [30]. The novel contributions of this paper are: (1) The definition of interaction mesh with per-edge weighting. (2) Describing the topological properties of the writhe scalar. (3) Introducing winding numbers as simplification of writhe coordinates. (4) Experiments demonstrating motion generalization using topology-based representations.

In the rest of this section, we will first review previous work on the use of topology-based representations for character animation and abstract spaces used for robot motion synthesis. Section 2 then introduces three specific types of alternate representations, two of which have their origins in topological properties of strings—the writhe—and the winding numbers and the third capturing the relative distances between interacting parts. We describe the implications of these representations in Section 2.1.2. Section 3 presents our approach to combining topology-based and configuration space representations in an optimal control based planning scheme implemented through the Approximate Inference Control (AICO) [27, 21]) framework – with their specific motion priors coupled via the graphical model. Section 4 addresses the topic of using these representations to generalize motion to novel situations by "remapping" it using this more abstract space. Finally, in Section 5 we present experiments on using the proposed methods to solve motion synthesis problems like unwrapping that are infeasible (e.g. for RRT methods) without exploiting alternate representations and demonstrate generalizability to novel, dynamic situations.

1.1 Previous Work

Controlling objects or robots with large degrees of freedom is a difficult problem. Previous methods solve such problems by either reducing the dimensionality of the problem by abstracting the state space or introducing a Voronoi bias into the sampling strategies of random exploration methods methods. We review some of these methods next, followed by a discussion on methods of implementing stochastic optimal control as a planner.

Dimensionality Reduction: There is a strong interest in reducing the dimensionality of the state space of robots to simplify the control strategy. Machine Learning techniques such as Gaussian processes have been successfully applied for such purposes[12]. In [3], a latent manifold in joint angle space is computed using Gaussian process from sample configurations produced by an expert. This manifold is, however, defined by samples from a valid trajectory in joint angle space and it does not capture the state of the environment directly. As

a result, when these methods are applied to problems of close interactions, artifacts such as penetrations start to appear. This problem cannot be resolved by simply making use of efficient collision detection (e.g., [10]) since qualitatively different movements such as avoiding obstacles from opposite sides may end up being blended.

Abstraction based on Spatial Relations: Another way to reduce the dimensionality of the state space is to explicitly specify the parameters for abstracting the state space. For example, in order to cope with problems of close interactions, we can represent the state space based on the spatial relations between the body parts and objects. Such representations can be found in knotting robots [25, 28, 15, 23]. that describe the status of the strands based on its overlap with itself, when viewing from a specific direction [5]. The obvious disadvantage of such representations is its view-dependence and the difficulty in generating commands to actually manipulate the elements, e.g., the rope.

Alternate topology-based representations that describe the relationships between 1D curves using their original configurations have been used for motion synthesis [8, 26] and classifying paths into homotopy groups [2]. Similarly, in [8], a representation called Topology Coordinates that is based on the Gauss Linking Integral has been proposed to synthesize tangling movements. The inverse mapping from the new coordinates to the joint angles are generated by the least squares principle using inverse kinematics. In [26], the same representation is applied for controlling the movement of a robot that puts a shirt on a human. Here, the inverse mapping is learned via human demonstrations. Such an approach is applicable only when the desired movement is consistent and simple, but not when the planning needs to be done between arbitrary sample points in the state space. An idea to abstract the paths connecting a start point and the end point using homotopy classes in 2D based on complex numbers [1] and in 3D based on the Ampere's law [2] has also been proposed. However, this work stops at classifying paths into homotopy groups and there is no recipe for mapping from the topology-based representation to the low level control coordinates. Moreover, these representations are only applicable for 1D curves and not for describing the relationship between 2D surfaces, which is case often needed for controlling robots.

Another representation called interaction mesh that describes the spatial relationship of the body parts is introduced in [9]. The relationship is quantified by the Laplacian coordinates of the volumetric mesh whose points are sampled on the body parts composing the scene. The advantage of this approach is that it is not restricted to describe the relationship between 1D curves but is extendable to those between 2D surfaces. However, it is a discrete representation which is only valid in the neighborhood of the posture from which the volumetric mesh is computed.

Exploratory Motion Planning Methods Exploratory motion planning such as Probabilistic Road Maps (PRM) and Rapidly exploring random trees (RRT) is another possible approach for controling systems with many degrees of freedom. As the complexity of such methods are exponential with respect to the degrees of freedom of the system, methods that introduce a Voronoi bias into the sampling strategy in the task space [24, 14] have been proposed. Sampling based planning that takes dynamics of the robot into account have been proposed [29]. The problem of planning in dynamic environments with multiple goals has been addressed in [7]. Such methods, however, do not fully utilize the modified metric induced by the alternate representations.

Stochastic Optimal Control: The field of stochastic optimal control has developed several techniques such as iLQG that have previously been used for optimising robot motion with

complex dynamics [13, 18, 4]. The stochastic optimal control problem can also be formulated as Approximate Inference Control (AICO) [27]. This method has also been used for time optimisation [20] and it is suitable for combining multiple representations by extending the graphical model. Our approach is to combine multiple criteria based on spatial relations using AICO.

In summary, using a representation based on the spatial relations is a promising direction to reduce the dimensionality of the control, although little work has been done for flexible path planning or optimal control in such state spaces.

2 Topology-based representations

The topology-based spaces we will introduce significantly modify the metric and topology of the search space. For instance, points that are near in the topology space might be far in configuration space, or a linear interpolation in topology space may translate to complex non-linear motion in configuration space. We are particularly interested in the change of topology, such that the new representation will capture invariants that enable us to represent interactions with the environment more effectively. The motion synthesis methods proposed in the next section are independent of the specific choice of representation. Here we will introduce three specific examples of topology-based representations (winding numbers, writhe coordinates and interaction mesh) which we will also use in the experiments. These representations have previously been used in the context of computer animation, namely the writhe representation [8] that captures the "windedness" of one object around another, and the interaction mesh representation [9] that captures the relative positioning of key points between interacting objects.

All three types of representations can be formalized by a mapping $\phi : q \mapsto y$ from configuration space to the topology-based space, where $q \in \mathbb{R}^n$ is the configuration state with n degrees of freedom. To be applicable within our motion synthesis system, we need to be able to compute ϕ and its Jacobian J for any q.

2.1 Writhe matrix and scalar

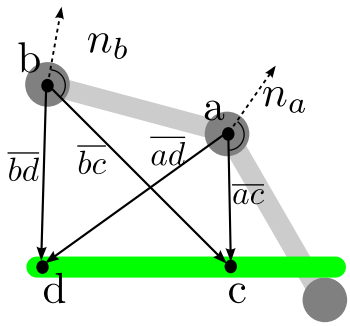


Figure 2: Illustration of the definition of writhe for two segments. One - ab - belongs to the manipulator and another - cd - is a part of the obstacle.

The writhe [11] is a property of the configuration of two kinematic chains (or in the continuous limit, of two strings). Intuitively the writhe describes to what degree (and how and where) the two chains are wrapped around each other, which is well suited for representing winding and wrapping motion.

Let us describe two kinematic chains by positions $p^{1,2}_{1:K}$ of their joints, where $p^i_k \in \mathbb{R}^3$ is the kth point of the ith chain. Using standard kinematics, we know how these points depend on the configuration $q \in \mathbb{R}^n$, that is, we have the Jacobian $J^i_k := \frac{\partial p^i_k}{\partial q}$ for each point. The writhe matrix is a function of the link positions $p^{1,2}_{1:K}$. More precisely, the **writhe matrix** W_{ij} describes the relative configuration of two points (p^1_i, p^1_{i+1}) on the first chain and two points (p^2_j, p^2_{j+1}) on the second where i, j are indexes of points along the first and the second chain respectively. For brevity, let us denote these points by $(a, b) = (p^1_i, p^1_{i+1})$ and $(c, d) = (p^2_j, p^2_{j+1})$, respectively (see Fig. 2). Then

$$W_{ij} = \left[\sin^{-1}\frac{n_a^\top n_d}{|n_a||n_d|} + \sin^{-1}\frac{n_b^\top n_c}{|n_b||n_c|} + \sin^{-1}\frac{n_c^\top n_a}{|n_c||n_a|} + \sin^{-1}\frac{n_d^\top n_b}{|n_d||n_b|} \right] \text{sign}\left[\overline{ab}^\top (\overline{ac} \times \overline{cd}) \right] \tag{1}$$

where n_a, n_b, n_c, n_d are normals at the points a, b, c, d with respect to the opposing segment (c.f. Fig. 2),

$$n_a = \overline{ac} \times \overline{ad}\ ,\ n_b = \overline{bd} \times \overline{bc}\ ,\ n_c = \overline{bc} \times \overline{ac}\ ,\ n_d = \overline{ad} \times \overline{bd}\ . \tag{2}$$

The above equations for computing the Writhe are an analytical expression for the Gauss linking integral along two segments. The solution of this integral is based on an analogy with the solid angle formed by all view directions in which segments (a, b) and (c, d) intersect [11] multiplied by an appropriate sign. Since the writhe matrix is a function of the link positions $p^{1,2}_{1:K}$ we can compute its Jacobian using the chain rule

$$\frac{\partial W_{ij}}{\partial q} = \frac{\partial W_{ij}}{\partial p^1_i} J^1_i + \frac{\partial W_{ij}}{\partial p^1_{i+1}} J^1_{i+1} + \frac{\partial W_{ij}}{\partial p^2_j} J^2_j + \frac{\partial W_{ij}}{\partial p^2_{j+1}} J^2_{j+1}\ . \tag{3}$$

Fig. 8 illustrates 2 configurations together with their writhe matrix representation. Roughly, the amplitude of the writhe (shading) along the diagonal illustrates which segments are wrapped around each other. From the full writhe matrix we can derive simpler metrics, usually by summing over writhe matrix elements. For instance, the Gauss linking integral, which counts the mean number of intersections of two chains when projecting from all directions, is the sum of all elements of the writhe matrix. In our experiments, we will also use the vector $w_j = \sum_i W_{ij}$ as a representation of the current configuration. Writhe, however, does not provide a unique mapping to joint angles which is why we usually require additional constraints and cost terms.

2.1.1 Winding numbers

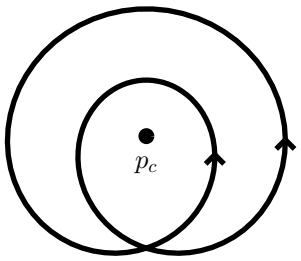


Figure 3: Winding number of a point p_c surrounded by the doubly wound curve: $\hat{w} = 2$.

If the problem can be simplified to planning a wrapping motion in 2D, we can use a special case of the writhe representation. The winding number defines how many times a curve is

wound around a point on a 2D plane (Fig. 3). We compute the Winding number using the approximate algorithm derived in [19] which is based on calculating inverse trigonometric functions of the scalar product of two normalized vectors, formed by consequent points p_i and p_{i+1} on a curve and a central point p_c:

$$\hat{w} = \frac{1}{2\pi} \sum_{i=1}^{n-1} \arccos \left(\frac{(p_i - p_c)^\top (p_{i+1} - p_c)}{|p_i - p_c||p_{i+1} - p_c|} \right) \tag{4}$$

where n is the number of points along the curve. This scalar continuous function can be thus viewed as a simplification of a writhe representation defined in Section 2.1. In our experiments that use the winding numbers, we assume that all joints and points lie within one plane. We use the standard kinematics to define the points p_i depending on the configuration $q \in \mathbb{R}^n$. We can therefore use the chain rule to compute the Jacobian of the winding number $\frac{\partial \hat{w}}{\partial q}$.

2.1.2 Homotopy classes of robot configurations

The writhe scalar is a topology invariant [6] based on homotopy that exists between configurations of the robot. Let us assume we have one kinematic chain p_k^i that depends on the robot configuration $q \in \mathbb{R}^n$ such as the skeleton of a robot arm. We define m as the number of chains p_k^j that do not depend on the configuration but interact with the robot such as skeletons of obstacles. We can now compute the writhe scalar w_{p^i,p^j} between chains p_k^i and p_k^j. If there exists a *homotopy* between arbitrary two configurations of the robot, we can control the robot to smoothly move between these two configurations without intersecting with the skeletons of the obstacles and we say that these configurations belong to the same homotopy class. *If two chains p^1 and p^2 describing two configurations of a robot belong to the same homotopy class and connect the same two points in space then the value of their respective writhe scalars is the same: $w_{p^1,p^j} = w_{p^2,p^j}$.* To prove this we concatenate these two chains to create a closed loop $p^1 \cup -p^2$. Here $-p^2$ indicates that the orientation of the curve is reversed. The writhe scalar of $w_{p^1 \cup -p^2,p^j}$ is non-zero if the closed loop encloses the chain p^j and it is zero otherwise. Therefore, the writhe scalars of the chains p^1 and p^2 must be equal if the chains do not enclose a third chain p^j describing the obstacle. This also means that one of the chains can be smoothly deformed into the other without intersecting p^j. This argument extends to m chains describing the obstacles *individually*. The consequences of using this representation for motion planning are: (1) We can constrain the planning to a particular homotopy class (reach inside a loop as described in Section 5.3). (2) Collision avoidance can be achieved by avoiding high values of writhe that occur near the intersecting configurations.

We call the writhe a topology-based representation because it is directly related to homotopy classes. The writhe scalar is, however, not a homotopy invariant but rather a homology invariant. Here we use homology as approximation of homotopy [22]. Homology is a weaker topology invariant which means that if two configurations are homotopic they are also homologous but not necessarily the other way around. As a result the statement: *"If two configurations of a kinematic chain have the same writhe scalar they belong to the same homotopy class."* is not true universally. An example of such scenario is shown in Fig. 4 where the configurations p^1 (solid black) and p^2 (dashed) have the same value of writhe scalar but they belong to different homotopy classes with respect to the two skeletons of the obstacles (circles). In a motion planning scenario, if the task was to reach the goal configuration while maintaining the end-effector position constraint, it would be necessary to break this constraint in order to reach the goal configuration, with a potential danger of local minima.

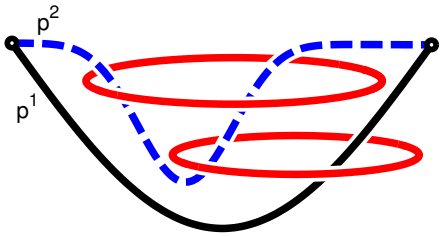


Figure 4: An example of two configurations of a kinematic chain represented by a solid black curve (p^1) and a dashed curve (p^2) that are homologous but not homotopic with respect to both obstacles (circles). As a result it is not possible to deform the black curve into the dashed one without intersecting at least one of the circles.

2.2 Interaction mesh

The writhe can only represent interaction between a pair of kinematic chains. The authors of [9] have, however, proposed a representation called interaction mesh that can be used to capture the spatial interaction of the robot with its environment. The interaction mesh is a function of a graph connecting a set of landmark points on the robot and in the environment. More precisely, assume we have a set of points $P = \{p_i\}_i$ including landmarks on a kinematic robot configuration and on objects in the environment. Let G be a (bi-directional fully or partially connected) graph on P. To each vertex $p \in G$ in the graph, we associate the Laplace coordinate

$$L_G(p) = p - \sum_{r \in \partial_G p} \frac{r w_{pr}}{\sum_{s \in \partial_G p} w_{ps}} \tag{5}$$

$$w_{pr} = \frac{W_{pr}}{|r-p|}, w_{ps} = \frac{W_{ps}}{|s-p|} \tag{6}$$

where $\partial_G p$ is the neighbourhood of p in the graph G and w_{pr} is the weight inversely proportional to the distance of points p, r and multiplied by the manually chosen edge importance weighting W_{pr}. The weights are then normalized over the neighbouring nodes in the graph $s \in \partial_G p$. The collection of Laplace coordinates of all points,

$$M = (L_G(p))_{p \in P} \ , \tag{7}$$

is a $3|P|$-dimensional vector which we denote as *interaction mesh*. As with the writhe matrix, we assume the Jacobian of all robot landmarks in P is given and the Jacobian of other environmental landmarks is zero. The Jacobian $\frac{\partial M}{\partial q}$ of the interaction mesh is given via the chain rule.

We would like to point out that the squared metric in M-space has a deformation energy interpretation [9]. To see this, consider a change of position of a single vertex p to a new position p'. The deformation energy associated to such a change in position is defined based on the neighbourhood in a tetrahedronisation T of the point set:

$$E_T(p') = \frac{1}{2} \| L_T(p') - L_T(p) \|^2 \tag{8}$$

where $L_T(p)$ are the Laplace coordinates of p w.r.t. the tetrahedronisation T. The difference to our definition of the interaction mesh is that we consider Laplace coordinates L_G w.r.t. the

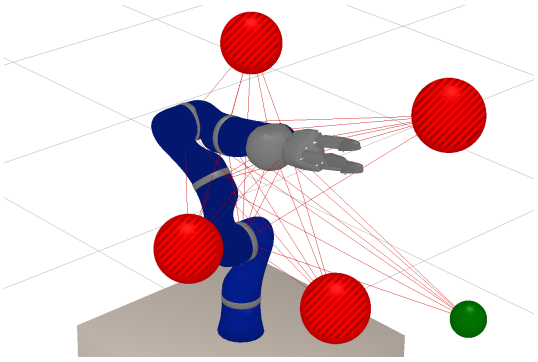


Figure 5: Interaction mesh created from points set P depending on positions of the obstacles (striped spheres), the goal (solid sphere) and the links of the KUKA LWR 4 arm. Edges of the interaction mesh are shown as lines connecting the points. Edges between the obstacles have been removed.

fully connected graph G instead of only T, which is a sub-graph of the fully connected graph G computed using tetrahedronisation of points P. Since different configurations lead to topologically different T, using L_G has the benefit of more continuous measures (deformation energies as well as Jacobian). Neglecting this difference, minimizing squared distances in M-space (as is implicit, e.g., in inverse kinematics approaches as well as the optimal control approaches detailed below) therefore corresponds to minimizing deformation energies. The choice of a particular graph connectivity G and the edge importance weights W_{pr} should reflect the desired interaction of the robot with the environment.

3 Optimal control combining topology-based and configuration space representations

The motivation for introducing topology-based spaces is that they may provide better metrics or topology for motion synthesis, ideally such that local optimization methods within topology-based space can solve problems that would otherwise require more expensive global search in configuration space. In this section, we describe our method for exploiting topology-based spaces for motion synthesis in an optimal control context. We formulate the approach within the framework of Approximate Inference Control [21], which is closely related to differential dynamic programming [17] or iLQG [13] (see details below), and will allow us to use a graphical model to describe the coupling of motion estimation on both representations. In the following Section, we first briefly introduce this framework before we explain how to couple additional representations in the probabilistic inference framework.

3.1 Approximate Inference Control

Approximate Inference Control (AICO) frames the problem of optimal control as a problem of inference in a dynamic Bayesian network. Let x_t be the state of the system—we will always consider the dynamic case where $x_t = (q_t, \dot{q}_t)$. Consider the problem of minimizing (the

expectation of) the cost

$$C(x_{0:T}, u_{0:T}) = \sum_{t=0}^{T} c_x(x_t) + c_u(u_t) \tag{9}$$

where c_u describes costs for the control and c_x describes task costs depending on the state (usually a quadratic error in some task space). The robot dynamics are described by the transition probabilities $P(x_{t+1} \,|\, u_t, x_t)$. The AICO framework translates this to the graphical model

$$p(x_{0:T}, u_{0:T}) \propto P(x_0) \prod_{t=0}^{T} P(u_t) \prod_{t=1}^{T} P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1}) \cdot \prod_{t=0}^{T} \exp\{-c_x(x_t)\} \,. \tag{10}$$

The control prior $P(u_t) = \exp\{-c_u(u_t)\}$ reflects the control costs, whereas the last term $\exp\{-c_x(x_t)\}$ reflects the task costs and can be interpreted as "conditioning on the tasks" in the following sense: We may introduce an auxiliary random variable z_t with $P(z_t = 1 \,|\, x_t) \propto \exp\{-c_x(x_t)\}$, that is, $z = 1$ if the task costs $c_x(x_t)$ are low in time slice t. The above defined distribution is then the posterior $p(x_{0:T}, u_{0:T}) = P(x_{0:T}, u_{0:T} \,|\, z_{0:T} = 1)$. AICO in general tries to estimate p, in particular the posterior trajectory and controls. In [27], this is done using Gaussian message passing (comparable to Kalman smoothing) based on local Gaussian approximations around the current belief model. In [21], theory on the general equivalence of this framework with stochastic optimal control is detailed. Generally, the approach is very similar to differential dynamic programming [17] or iLQG [13] methods with the difference that not only backward messages or cost-to-go functions are propagated but also forward messages ("cost-to-reach functions"), which allows AICO to compute a local Gaussian belief estimate $b(x_t) \propto \alpha(x_t)\beta(x_t)$ as the product of fwd and bwd message and utilize it to iterate message optimization within each time slice.

3.2 Expressing motion priors in topology-based spaces and coupling spaces

To estimate the posterior, the controls u_t can be marginalized, implying the following **motion prior**:

$$P(x_{t+1} \,|\, x_t) = \int_u P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1})\; P(u_t)\; du \,. \tag{11}$$

This motion prior arises as the combination of the system dynamics and our choice of control costs $c_u(u_t)$ in x-space; for LQ systems it is a linear Gaussian.

The motion prior is a unique view on our motivation for topology-based representations. In the introduction, we mentioned the impact of representations on the Voronoi bias, the metric, or the topology. In other terms, successful trajectories are likely to be "simpler" (easier to find, shorter, local) in an appropriate space. In Machine Learning terms, this is expressed in terms of a prior. In this view, topology-based spaces are essentially a means to express priors about potentially successful trajectories—in our case we employ the linear Gaussian prior in a topology-based space to express the belief that trajectories may appear "simple" in a suited topology-based space.

However, using AICO with a linear Gaussian motion prior in topology space is not sufficient to solve general motion synthesis problems: 1) The computed posterior in topology

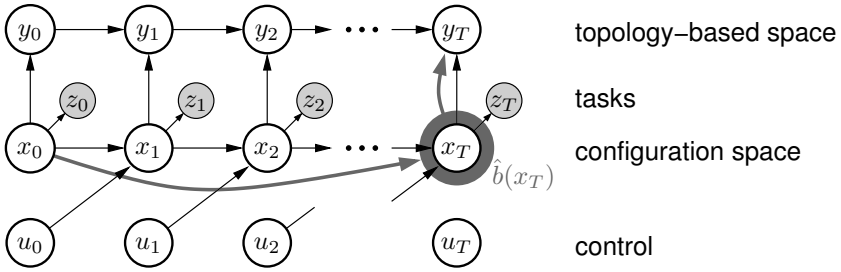


Figure 6: AICO in configuration and topology-based space. The grey arcs represent the approximation used in the end-state posterior estimation.

space does not directly specify an actual state trajectory or control law on the joint level. 2) We neglect the problem of minimization of control and task costs originally defined on the joint level. To address these issues, we need mechanisms to couple inference in topology-based and state space. We do so by coupling topology-based and joint state representations in AICO's graphical model framework.

Fig. 6 displays a corresponding graphical model. The bottom layer corresponds to the standard AICO setup, with the motion prior $P(x_{t+1} \,|\, x_t) = \int_u P(x_t \,|\, u_{t-1}, x_{t-1})\, P(u_t)\, du$ implied by the system dynamics and control costs. Additionally it includes the task costs represented by $P(z_t = 1 \,|\, x_t) = \exp\{-c_x(x_t)\}$. The top layer represents a process in topology-based space with an a priori given linear Gaussian motion prior $P(y_{t+1}|y_t)$. Both layers are coupled by introducing additional factors

$$f(x_t, y_t) = \exp\{-\frac{1}{2}\rho \|\phi(q_t) - y_t\|^2\}, \tag{12}$$

which essentially aim to minimize the squared distance between the topology-based state y_t and the one computed from the joint configuration $\phi(q_t)$, weighted by a precision constant ρ. Note that for Gaussian message passing between levels using a local linearisation of ϕ (having the Jacobian of the topology-based space) is sufficient. These factors essentially treat the topology-based state y_t as an additional task variable for the lower level inference, analogous to other potential task variables like end-effector position or orientation. In our setup, we generally distinguish between configuration space, task space and topological space. While the task space generally describes a space sufficient to describe cost or rewards [1] the role of the topological space is to provide alternative metrics (and topology) for trajectory optimization. In our experiments, we build only two layers of hierarchy between the configuration space and the topology-based spaces (such as writhe coordinates and interaction mesh space). Since these spaces provide simple priors that capture the task well there is no need for a more structured graphical model. It is, however, possible to build more complex hierarchies by extending the graphical model.

The choice of topology-based spaces and their weighting depends on the task — in our experiments, these are chosen manually. For example the winding numbers are suitable for tasks where the expected motion is a winding motion that can be projected on a 2D plane (see Section 5.1). If 3D winding is required and both the robot and the skeletons of the obstacles can be represented by kinematic chains, then writhe space should be used (see Section 5.2). When preserving local interactions, between the robot and the environment is necessary, the

[1] More precisely, we may define a task space as a projection of configuration space for which we apriori know that costs (other than transition costs) depend only on this task space.

interaction mesh should be used. It is, however, possible to arbitrarily combine these representations if, for example, the task requires both preserving the nature of interaction as well as planning a wrapping motion (see Section 5.3).

3.3 End-state posterior estimation

Probabilistic inference in the full factor graph given in Fig. 6 would require joint messages over configuration and topology-based variables or loopy message passing. We approximate this inference problem by a stage-wise inference process:

1. We first focus on approximating directly an end-state posterior $\hat{b}(x_T)$ for the end-state x_T which fulfils the task defined in configuration space. We explain the method used for this below.
2. Accounting for the coupling of this end-state posterior to the topology-based representation, we compute a trajectory posterior in topology-based space.
3. We then project this down to the joint level, using AICO in configuration space coupled to the topology-based space via factors introduced above.

Clearly this scheme is limited in that the initial inference in topology space only accounts for the task at the final time step. To overcome this limitation, we would have to iterate inference between levels. For the problems investigated in our experiments, the approximation scheme above is sufficient.

End-state posterior estimation computes an approximate belief $\hat{b}(x_T) \approx P(x_T \mid x_0, z_{0:T}=1)$ about the final state given the start state and conditioned on the task. This approximation neglects all intermediate task costs and assumes linear Gaussian system dynamics of the form

$$P(x_t \mid x_{t\text{-}1}) = \mathcal{N}(x_t \mid A_t x_{t\text{-}1} + a_t, W_t) \,. \tag{13}$$

We integrate the system dynamics,

$$P(x_T \mid x_0) = \sum_{x_{1:T\text{-}1}} \prod_{t=1}^{T} P(x_t \,|\, x_{t\text{-}1}) \,, \tag{14}$$

which corresponds to the grey arc in Fig. 6. For stationary linear Gaussian dynamics, we have

$$P(x_T|x_0) = \mathcal{N}(x_T|A^T x_0 + \sum_{i=0}^{T\text{-}1} A^i a, \sum_{i=0}^{T\text{-}1} A^i W A^{'i}) \,, \tag{15}$$

where superscript on A stands for a power of matrix, defined iteratively $A^i = A * A^{i-1}$. To estimate $\hat{b}(x_T)$, we condition on the task,

$$P(x_T \mid x_0, z=1) = \frac{P(z_T=1 \mid x_T)\, P(x_T \mid x_0)}{P(z_T=1 \mid x_0)} \,. \tag{16}$$

Since we assume $P(x_T \mid x_0)$ to be Gaussian, using a local Gaussian approximation of the task $P(z_T=1 \mid x_T)$ around the current mode of $\hat{b}(x_T)$, $P(x_T \mid x_0, z=1)$ can be approximated with a Gaussian as well. We iterate this by alternating between updating the end-state estimate $\hat{b}(x_T)$ and re-computing the local Gaussian approximation of the task variable. A Levenberg-Marquardt type damping (depending on monotonous increase of likelihood) ensures robust and fast convergence.

4 Generalisation and re-mapping using a topology-based space

The optimal control approach presented in the previous section exploits the additional topology-based space to generate an optimal trajectory (and controller). However, the computed optimal trajectories may no longer be valid when there is a change in the environment, e.g. the obstacles have moved. In order to cope with this issue, we need a dynamic replanning method. The topology-based representations are invariant to certain changes in the environment. The replanning at the topology level is therefore not necessary when the representation generalizes the desired motion. We propose a per-frame re-mapping approach in which the optimal trajectory in the topology-based representation $y^*_{0:T}$ is inverse-mapped to the configuration space according to the novel condition of the environment.

Technically, this is done by computing the configuration of the system per-frame such that the remapping error from the original optimal topology-based trajectory $y^*_{0:T}$ is minimized. However, topology-based representations such as the writhe matrix or the interaction mesh are very high-dimensional—often higher dimensional than the configuration space itself. This is in strong contrast to thinking of $y^*_{0:T}$ as a lower dimensional task space like an end-effector space. Therefore, following $y^*_{0:T}$ exactly is generally infeasible and requires a regularisation procedure that minimizes the 1-step cost function:

$$f(q_{t+1}) = \|q_{t+1} - q_t - h\|^2 + \|\phi(q_{t+1}) - y^*\|^2_C \,, \tag{17}$$

$$\underset{q_{t+1}}{\operatorname{argmin}} f(q_{t+1}) = q_t + J^\sharp(y^* - y_t) + (I - J^\sharp J)h \tag{18}$$

$$\text{with } J^\sharp = J^\top(JJ^\top + C^{-1})^{-1}$$

where C describes a cost metric in y-space.

For the case of the interaction mesh, we mentioned the relation of a squared metric C in M-space to the deformation energy. Therefore, using the per-frame remapping to follow an interaction mesh reference trajectory $M^*_{0:T}$ essentially tries to minimize the deformation energy between the reference M^*_t and the actual $\phi(q_t)$ at each time step. This implies generalizing to new situations by approximately preserving relative distances between interacting objects instead of directly transferring joint angles. In conjunction with the use of feedback gains, the methodology proposed here is able to cope with dynamic environments (see Section V-C) and bounded unpredictable changes.

The bottleneck for a feedback controller using this methodology is forwards mapping of the topology-based representations. The forward mapping of the winding numbers has the computational complexity of $O(n)$, n being the number of linear segments approximating the curve being wound. Computing the writhe coordinates requires $O(nm)$ number of operations (each operation is defined by Equation 1), where n and m are number of segments of the two kinematic chains respectively. Computing the interaction mesh has the computational complexity of $O(2n)$ where n is the number of vertices of the mesh. The performance is therefore mainly defined by granularity of the approximation of the geometry of the robot and the environment. The experiment in Section 5.2 shows that computation overhead (e.g. the average running time is about 10 seconds on standard 4 core 2.40GHz computer with 4 GB memory) becomes affordable when the task is relatively complex. On the other hand, the experiments in Section 5.3 and Section 5.4 show that an interaction mesh with a small number of vertices can be used in a feedback loop to provide real time control.

5 Experiments

5.1 Paper folding using winding numbers

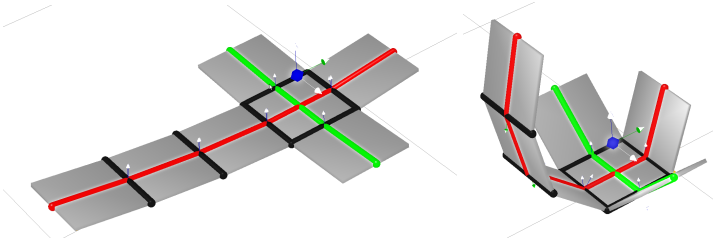


Figure 7: Initial (left) and final (right) state of simulation of box folding using winding numbers representation. The two skeletons that are being wound around the centre of the box are overlaid over the sides of the box.

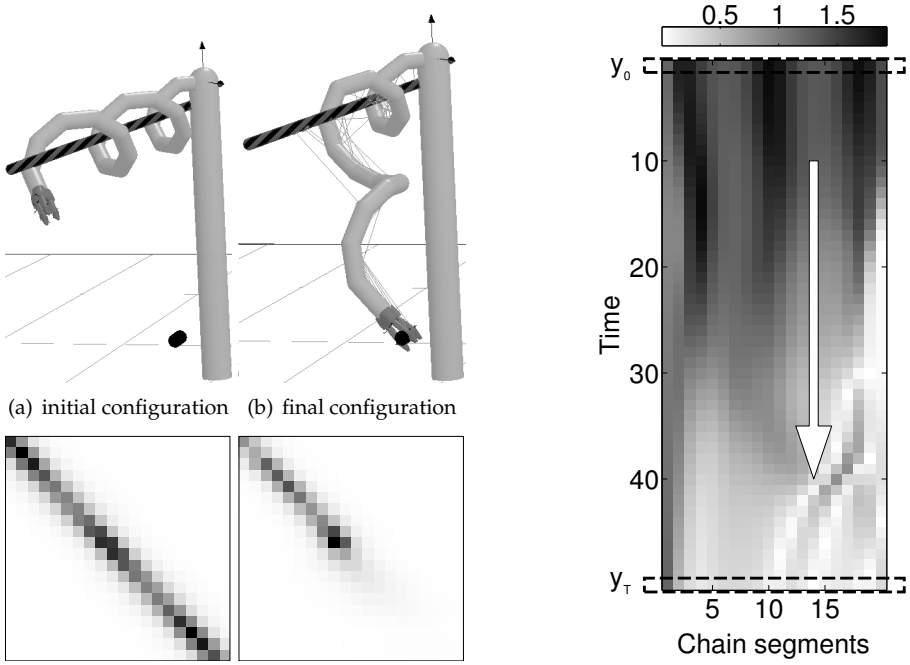
Our first experiment is a toy example of how topology-based representations can be used to solve the planning problem. Being the most intuitive topological representation, the winding number essentially captures a degree of windedness of a chain around a central point. In order to demonstrate the elegance and simplicity of this abstraction, we simulated a folding a box from layout experiment (See Fig. 7). Instead of controlling all joint angles directly (although trivial in this case), we set only two scalar goals for our optimization problem — degrees of twist for the two chains. Close up or folding motion is then achieved by the same algorithm as in other experiments — approximate inference in dynamic Bayesian network. Winding number has good generalization properties (e.g. it is invariant w.r.t. number of elements in chain) and can be used in arbitrary combination with other topological or geometrical goals. The video of this experiment is included in Extension 1.

5.2 Manipulator unwrapping and reaching using writhe space

As an example of a possible application of the writhe space abstraction, we simulated a manipulator, consisting of 20 segments and a hand with three fingers, making in total 29 DoF. Initially this rope-like manipulator is twisted two and a half times around a striped pole, giving us approximately 900° of writhe density (See Fig. 8(a)).

The task is to plan a trajectory which should grasp the black cylinder without colliding with the striped stick (Fig. 8(a),8(b) and video included in Extension 1). Clearly, a local feedback approach using Inverse Kinematics will experience failure in this task. As a result AICO with end-effector task variable and collision avoidance task variable is unable to converge to a solution due to a deep local minima in this space. To solve this planning problem in end-effector space, we have to use exploratory motion planning methods such as RRTs. On the other hand, a successful trajectory can be well captured as a linear interpolation in writhe space and projected back to the configuration space using the coupling described in Section 2.1. Fig. 8(e) illustrates an example of a unwrapping trajectory in topology-based space when all rows of writhe matrix are summed up into one column, representing the current state.

Hierarchical AICO conditioned on the end-state in writhe space y_T was able to generate locally optimal trajectories, consisting of 50 time steps, in only few iterations, requiring a relatively small number of expensive collision checks (less than 1000). Comparison with Rapidly-exploring Random Tree (RRT) planning for this reaching task revealed a dependence of the performance on distance between end-effector and object position. Moreover, total costs of



(a) initial configuration (b) final configuration

(c) initial writhe matrix (d) final writhe matrix (e) writhe trajectory

Figure 8: The experimental task is to grasp the object without collisions. Corresponding writhe matrices (c, d) are depicted below the configurations (a, b) - the darkness represents the amplitude of the writhe value. Each row in writhe space evolves over time as shown in (e).

obtained trajectories were on average 100 times higher than those generated with the local optimizer. Here, the end-state in configuration space q_T was given as a target for RRTs.

For more systematic evaluation of our planning platform, we have designed a sequence of unwrapping trajectories - gradually increasing the relative angle to be unwrapped. This sequence of final states was given as goals to uni- and bi-directional RRT planners. The results demonstrate that for simple trajectories (e.g. in case of nearby lying objects) all methods have no difficulties, whereas starting with one and a half of full twist, unidirectional search fails and bi-directional significantly slows down. (See Fig. 9.)

In this comparison, the RRTs solved a somewhat simpler problem than our system: For the RRTs we assumed to know the final state q_T in configuration space – we take our final pose estimate from Section 3.3 as the target q_T for RRTs. This is in contrast to our planning platform, where we use the final pose estimate only to estimate a final topology-based state y_T and then use the hierarchical AICO to compute an optimal trajectory (including an optimal q_T) conditioned on this final topology-based state. Therefore, the RRT's problem is reduced to growing to a specific end state. We applied the standard method of biasing RRT search towards q_T by growing the tree 10% of the time towards q_T instead of a random sample of the configuration space. Knowing q_T also allowed us to test bi-directional RRTs, each with 10% bias to grow towards a random node of the other tree. Even under such a simplified condition, the RRT-based planners require significantly more computation for interpolating the configurations when complex writhing is required. Further, RRTs output non-smooth paths whereas AICO produces (locally) optimal dynamic trajectories since it minimizes dynamic control costs. [2]

[2] AICO can also be applied on the kinematic level, where $x_t = q_t$ and with the control $u_t = q_{t+1} - q_t$ being just the joint angle step. This would reduce the computational cost further.

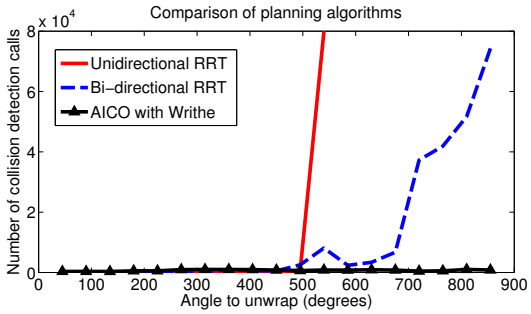


Figure 9: Performance of planning algorithms for unwrapping task. Computational time is proportional to the number of collision detection calls.

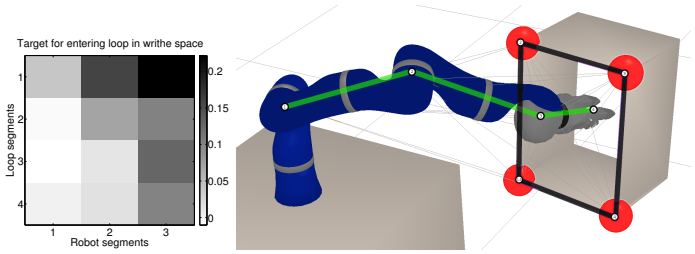
5.3 Dynamic reaching through a loop using writhe and interaction mesh

Writhe space is a suitable representation for tasks that involve interactions with chains—or loops—of obstacles. We have altered the task from Section 5.2 to reaching through a hollow box, where the rim of the box forms a loop of segments, see Fig. 10(b). Classically this problem would be addressed by exploiting the end-effector position collision avoidance. This is, however, a classical example of a bug trap problem in configuration space. The advantage of using writhe as a description of the interaction is in defining the task as a *relative* configuration of the robot and the loop—this relative description remains effective also when the box is moved dynamically. The writhe matrix corresponding to the final configuration peaks around the last link which passes through the box (see Fig. 1 and 10). This target in writhe space does not uniquely define the task for all arbitrary positions of the box (unlike the unwrapping task in Section 5.2) which allows for defining sub-goals such as precisely controlling the endeffector position via another task variable. We can therefore achieve accurate manipulation within a spatially constrained dynamic environment.

In this demonstration we use interaction mesh representation in addition to writhe coordinates which is well suited to represent reaching movement while maintaining relative positions of robot links w.r.t. the obstacles. In this case, the explicit collision avoidance was then superfluous. We coupled all three representations—writhe, interaction mesh, and joint configuration space—using AICO (see Equation 12) by extending the graphical model to efficiently generate motions for varying positions of the obstacles.

The computation of the optimal trajectory using this methods required 3 to 6 AICO iterations on average. We then used the same target in writhe space and randomly displaced the hollow box. Our method was able to plan collision-free trajectories for all test scenarios. We compared our method with AICO using only classical representations: end-effector position and collision avoidance. This required 20 to 30 AICO iterations on average and in 90% of the trials the algorithm failed to find a collision-free path. The remaining 10% of trials were successful because the the box was placed in a position where the bug trap problem did not manifest, e.g. the open side of the box was facing the robot.

We also tested online remapping as described in Section 4 using both the writhe and interaction mesh space to test behavior of the system when the box position is changed dynamically on the fly. For this, the position of a hollow box defining the loop was tracked using magnetic motion tracking system. We initially recorded the full trajectory in the topology-based space in a static environment. We then dynamically updated the robot's position in real time. We were able to reach inside of the hollow box without any collisions even when the box was moving. The video of this experiment is included in Extension 1.



(a) (b)

Figure 10: (a) Writhe space target for passing the last link of a robotic arm through a loop. (b) The configuration corresponding to the target in Writhe space. The loop was built by connecting the corners of the rim of the hollow box. The chains representing the robot and the obstacle are overlaid.

5.4 Motion generalisation using interaction mesh space

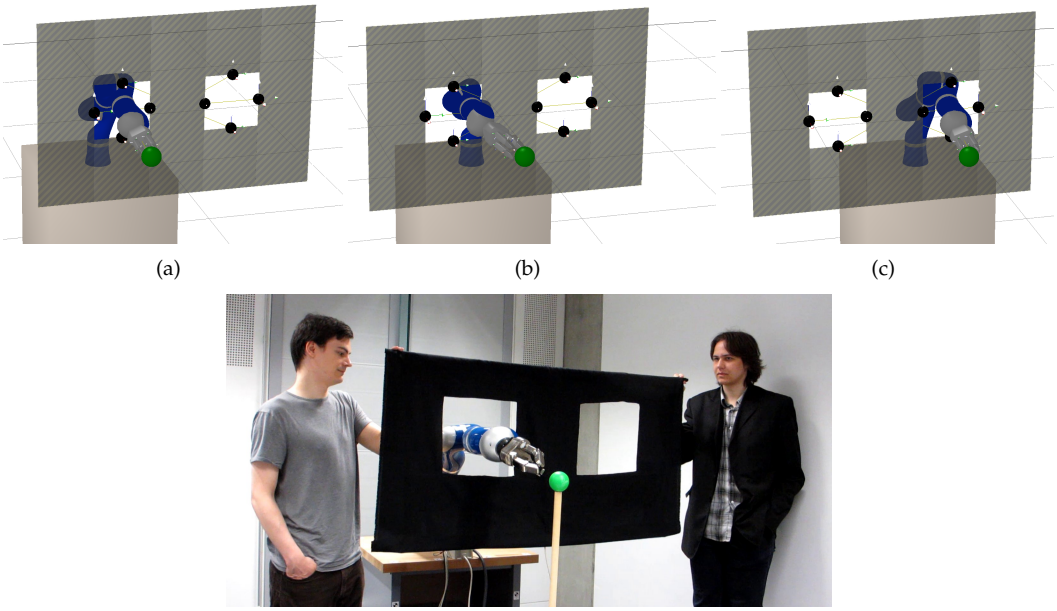
Finally, we present an experiment in which we show examples of motion generalisation when using topology-based representations. We used the KUKA LWR4 arm with 7 DOF and created a scenario common for factory environment where the arm is supposed to reach between items moving on a conveyor belt. The obstacle was a wall with two windows moving in front of the robot and obstructing the path to the goal (See Fig. 11(d)). We computed the initial trajectory in the static environment using AICO and the classical representations including end-effector position and collision avoidance. These representations are suitable for the simplified case where one of the windows is located in front of the robot. We used a forward mapping as defined in Eq. 6 to compute the reference trajectory in interaction mesh space.

We then used the remapping technique described in Section 4 to update the motion in real time while still fulfilling the task. Fig. 11(b) shows that the interaction mesh generalizes the motion well. When the wall moves to the left, the path eventually gets obstructed and the task cannot be fulfilled any more. We detected this by measuring the distance of the end-effector from the goal and using collision detection. In this case, re-planning is necessary. Fig. 11(c) shows the result of replanning.

In the second part of this experiment, we replaced the KUKA LWR4 arm with generic 7DOF manipulator with different kinematic structure. In order to demonstrate that plans in topology space can generalize across kinematic differences, we have manually defined correspondences of the landmarks points between the KUKA LWR4 and the generic manipulator. Fig. 12 shows that the result of re-using the plan in topology space with a modified inverse mapping. The higher DOF manipulator is able to also perform the task when the wall/opening is moving. The video of this experiment is included in Extension 1.

6 Conclusions

Different motion representations have different strengths and weaknesses depending on the problem. For certain interaction problems, there exist suitable topology-based representations in which the interaction can be described in a way that generalizes well to novel or dynamic situations (as with the interaction mesh), or where local optimization methods can find solutions that would otherwise require inefficient global search (as with the writhe representations). However, considering motion planning only in a topology-based representation is insufficient in order to additionally account for tasks and constraints in other representations.



(a) (b) (c)

(d)

Figure 11: (a) Initial plan for the robot arm reaching for a goal (sphere) avoiding the wall (striped). (b) Remapping the interaction mesh trajectory after moving the wall to the left. (c) Replanning when the generalisation fails due to the wall being moved too far from its original position. (d) Real time remapping on the real robot.

Previous work with such representations has only tested basic approaches for inverse mapping of fixed topological trajectories to the joint configuration [5, 8, 26]. In contrast, in this paper, we presented methods that combine the different representations at the abstract and lower level for motion synthesis. For instance, the reaching task in an end-effector space is coupled with a writhe space that allows a local optimization method to generate an unwrapping-and-reaching motion. Considering such a problem solely in joint configuration and end-effector space leads to "deep local minima" that are practically infeasible to solve—as our comparison to RRTs in Section 5.2 demonstrated. Considering such a problem only in writhe space would not address the actual reaching task.

We chose to formulate our approach in the framework of optimal control as an approximate inference problem since this allows for a natural extension of the graphical model to incorporate multiple representations. Alternative formulations are possible, for instance as a structured constraint optimization problem (MAP inference in our graphical model) that could be solved by methods such as SNOPT. What we coined as a motion prior in topology-based spaces would here correspond to pseudo control costs for transitions in topology-based space. Which formulation will eventually lead to computationally most efficient algorithms is a subject of active research. As an outlook, we aim to apply the proposed methods for dexterous robot manipulation (including grasping) of more complex, articulated or flexible objects, where we believe that multiple parallel representations will enable more robust and generalizable motion synthesis strategies.

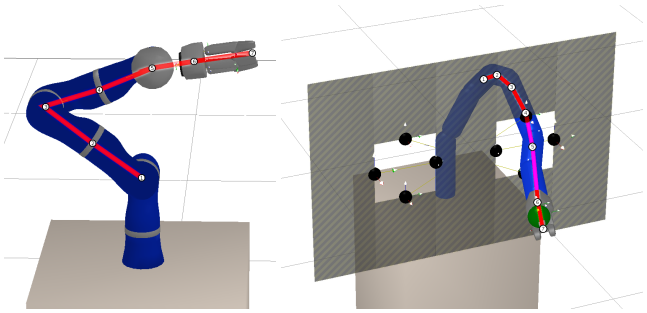


Figure 12: (left) Kinematic chain of the KUKA LWR4 robot used for planning. (right) Remapping the reaching trajectory onto robot with different kinematics. The points used for defining the interaction mesh and the corresponding part of the kinematic chain of the robot are overlaid.

Acknowledgements

This work was funded by the EU Seventh Framework Programme (FP7) as part of the TOMSY project. Prof. S. Vijayakumar is supported through a Microsoft Research Royal Academy of Engineering senior research fellowship.

References

[1] S. Bhattacharya, V. Kumar, and M. Likhachev. Search-based path planning with homotopy class constraints. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, Atlanta, GA, USA, 2010. AAAI Press.

[2] S. Bhattacharya, M. Likhachev, and V. Kumar. Identification and representation of homotopy classes of trajectories for search-based path planning in 3D. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, 2011.

[3] S. Bitzer and S. Vijayakumar. Latent spaces for dynamic movement primitives. In *Proc. of 9th IEEE-RAS International Conference on Humanoid Robots*, pages 574–581. IEEE, 2009.

[4] David Braun, Matthew Howard, and Sethu Vijayakumar. Optimal variable stiffness control: formulation and application to explosive movement tasks. *Autonomous Robots*, 33:237–253, 2012. 10.1007/s10514-012-9302-3.

[5] C. H. Dowker and B. T. Morwen. Classification of knot projections. *Topology and its Applications*, 16(1):19–31, 1983.

[6] H. Edelsbrunner and J. Harer. *Computational Topology: An Introduction*. AMS Press, 2010.

[7] R. Gayle, K. R. Klingler, and P. G. Xavier. Lazy reconfiguration forest (lrf) - an approach for motion planning with multiple tasks in dynamic environments. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 1316–1323, Rome, Italy, 2007. IEEE.

[8] E. S. L. Ho, T. Komura, S. Ramamoorthy, and S. Vijayakumar. Controlling humanoid robots in topology coordinates. In *Proc. of 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 178–182, Taipei, Taiwan, 2010. IEEE.

[9] E. S. L. Ho, T. Komura, and Ch. Tai. Spatial relationship preserving character motion adaptation. *ACM Transactions on Graphics*, 29:1–8, 2010.

[10] T. C. Hudson, M. C. Lin, J. D. Cohen, S. Gottschalk, and D. Manocha. V-COLLIDE: Accelerated collision detection for VRML. In *Proceedings of the second symposium on Virtual reality modeling language*, pages 117–ff., Monterey, CA, USA, 1997. ACM.

[11] K. Klenin and J. Langowski. Computation of writhe in modeling of supercoiled DNA. *Biopolymers*, 54(5):307–317, 2000.

[12] N. D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *Proc. of Neural Information Processing Systems (NIPS)*, Vancouver, B.C., Canada, 2003.

[13] W. Li and E. Todorov. An iterative optimal control and estimation design for nonlinear stochastic system. In *Proc. of the 45th IEEE Conference on Decision and Control*, pages 3242–3247, San Diego, CA, USA, 2006. IEEE.

[14] S. R. Lindemann and S. M. LaValle. Incrementally reducing dispersion by increasing voronoi bias in RRTs. In *Proc. of International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3251–3257, 2004.

[15] T. Matsuno and et.al. Manipulation of deformable linear objects using knot invariants to classify the object condition based on image sensor information. *IEEE/ASME Transactions on Mechatronics*, 11, 2006.

[16] J. R. Munkres. *Topology*. Prentice Hall, Incorporated, 2000.

[17] D. M. Murray and S. J. Yakowitz. Differential dynamic programming and Newtons method for discrete optimal control problems. *Journal of Optimization Theory and Applications*, 43:395–414, 1984.

[18] J. Nakanishi, K. Rawlik, and S. Vijayakumar. Stiffness and temporal optimization in periodic movements: An optimal control approach. In *Proc. of 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 718–724, San Francisco, CA, USA, 2011. IEEE.

[19] J. O'Rourke. *Computational Geometry in C*. Cambridge University Press, New York, NY, USA, 1998.

[20] K. Rawlik, M. Toussaint, and S. Vijayakumar. An approximate inference approach to temporal optimization in optimal control. In *Proc. of Neural Information Processing Systems (NIPS)*, Vancouver, B.C., Canada, 2010.

[21] K. Rawlik, M. Toussaint, and S. Vijayakumar. On stochastic optimal control and reinforcement learning by approximate inference. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, 2012.

[22] J. J. Rotman. *An Introduction to Algebraic Topology*. Springer-Verlag, 1988.

[23] M. Saha and P. Isto. Manipulation planning for deformable linear objects. *IEEE Transaction on Robotics*, 23:1141–1150, 2007.

[24] A. Shkolnik and R. Tedrake. Path planning in 1000+ dimensions using a task-space Voronoi bias. In *Proc. of 2009 IEEE International Conference on Robotics and Automation*, pages 2892–2898, Kobe, Japan, 2009. IEEE.

[25] J. Takamatsu and et.al. Representation for knot-tying tasks. *IEEE Transactions on Robotics*, 22:65–78, 2006.

[26] T. Tamei, T. Matsubara, A. Rai, and T. Shibata. Reinforcement learning of clothing assistance with a dual-arm robot. In *Proc. of 11th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 733–738, Bled, Slovenia, 2011. IEEE.

[27] M. Toussaint. Robot trajectory optimization using approximate inference. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, pages 1049–1056, New York, NY, USA, 2009. ACM.

[28] H. Wakamatsu, E. Arai, and S. Hirai. Knotting/unknotting manipulation of deformable linear objects. *International Journal of Robotics Research*, 25(4):371–395, 2006.

[29] E. Yoshida, C. Esteves, I. Belousov, J. P. Laumond, T. Sakaguchi, and K. Yokoi. Planning 3-D Collision-Free Dynamic Robotic Motion Through Iterative Reshaping. 24:1186–1198, 2008.

[30] D. Zarubin, V. Ivan, M. Toussaint, T. Komura, and S. Vijayakumar. Hierarchical Motion Planning in Topological Representations. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, 2012.

7 Appendix A: Index to Multimedia Extensions

The multimedia extensions to this article can be found online by following the hyperlinks from www.ijrr.org.

Extension	Type	Description
1	Video	Experiments demonstrating the advantages of using topology-based representations for motion planning.